Wikipedia Search Engine using various Embedding Techniques (Using BERT)

- **Updated problem formulation and literature review:**

➢ **Updated problem Statement:** Project aims to build a Wikipedia search engine to provide the relevant articles based on the user search query. The search engine will use BERT to generate embeddings for the input query and for the Wikipedia articles, which will be used to compute the relevance of the articles to the query. The Performance of the search engine will be evaluated based on the given metrics - precision, recall, F1 score, MAP(Mean Average Precision), MRR(Mean Reciprocal Rank) and accuracy of the retrieved articles.

➢ **Literature Review:**

○ **"On Improving Wikipedia Search using Article Quality"** by Meiqun Hu et al. which proposed a framework that re-ranks search results based on article quality based on two development quality measurement models, namely Basic and PeerReview, based on co-authoring data gathered from articles' edit history.
The authors begin by discussing the difficulties associated with Wikipedia search, such as the presence of low-quality articles and a lack of user feedback. They then propose a novel approach to ranking search results that uses article quality as a relevance criterion.
The proposed method is divided into two stages: the first is concerned with identifying high-quality articles, while the second is concerned with ranking the search results based on their relevance and article quality. Article quality is determined by a number of factors, including article length, edit frequency, and user ratings.
The authors evaluate their approach using a large dataset of Wikipedia articles and user queries. The results show that incorporating article quality information into the search process improves the effectiveness of Wikipedia search, particularly in retrieving high-quality articles.
The paper makes an important contribution to the field of information retrieval by demonstrating the usefulness of incorporating article quality information into the search process. The proposed method could be applied to other large-scale collaborative knowledge bases like Wikidata and DBpedia. However, the paper does not provide a detailed analysis of the proposed approach's limitations or generalizability to other contexts. Overall, the paper offers useful insights into how to improve the effectiveness of Wikipedia search by taking article quality information into account.
Link - https://dl.acm.org/doi/pdf/10.1145/1316902.1316926

○**"Leveraging BERT for Extractive Text Summarization on Lectures"** by Derek Miller which proposes a mechanism for extractive summarization through the use of K-Means clustering algorithm and the BERT model for text embeddings to find the closest sentences.
The paper starts by describing the challenges that come with summarizing lecture transcripts, such as the length and complexity of the transcripts, as well as the need to capture important concepts and ideas. The authors then propose a BERT-based method for identifying and ranking important sentences in a transcript using contextualized word embeddings.
The proposed approach is divided into three stages: the first involves pre-processing the transcript and encoding it with BERT; the second involves using a sentence scorer to identify important sentences in the transcript based on their contextualized embeddings; and the third stage involves selecting the top-ranked sentences to generate the summary.
The authors compare their approach to several baseline methods using a dataset of lecture transcripts. The findings show that the proposed method outperforms the baseline methods in terms of ROUGE (Recall-Oriented Understudy for Gisting Evaluation) scores as well as human evaluations.
The paper contributes significantly to the field of text summarization by demonstrating the efficacy of using BERT for extractive summarization of lecture transcripts. The proposed method could be applied to other domains involving complex and technical text, such as scientific articles and legal documents. However, the paper does not provide a detailed analysis of the proposed approach's limitations or generalizability to other contexts. Overall, the paper provides useful information on how to leverage BERT for extractive text summarization.
Link - https://arxiv.org/ftp/arxiv/papers/1906/1906.04165.pdf

○**"A survey on word embedding techniques and semantic similarity for paraphrase identification"** by Divesh R. Kubal, Anant V. Nimkar aims to provide an analysis of existing similarity metrics, statistical machine translation metrics and includes different word embedding techniques, stepwise derivation of its learning module.
The authors compare their approach to several baseline methods using a dataset of lecture transcripts. The findings show that the proposed method outperforms the baseline methods in terms of ROUGE (Recall-Oriented Understudy for Gisting Evaluation) scores as well as human evaluations.
The paper contributes significantly to the field of text summarization by demonstrating the efficacy of using BERT for extractive summarization of lecture

transcripts. The proposed method could be applied to other domains involving complex and technical text, such as scientific articles and legal documents.
The authors then use a dataset of paraphrases to compare the performance of various similarity metrics and statistical machine translation metrics. The findings show that word embedding-based methods outperform traditional similarity metrics and statistical machine translation metrics in most cases.
The paper contributes significantly to the field of natural language processing by providing a thorough analysis of existing similarity metrics and statistical machine translation metrics for paraphrase identification, as well as an overview of various word embedding techniques and their learning modules. The paper is well-organized and gives a thorough overview of the various techniques and methods. Overall, the paper offers useful insights into how to use word embeddings for paraphrase identification.
Link - https://www.inderscienceonline.com/doi/pdf/10.1504/IJCSYSE.2019.098417

○**"NLP based Intelligent News Search Engine using Information Extraction from e-Newspapers"** by Monisha Kanakaraj, Sowmya Kamath S present a personalized news search engine that focuses on building a repository of news articles by applying efficient extraction of text information using TF-IDF from a web news page from varied e-news portals.
The paper first provides an overview of the challenges associated with news search, such as the need to accurately identify relevant articles and extract key information from them. The authors then propose a search engine that extracts information from e-newspapers and creates a database of relevant articles using NLP techniques.
The proposed search engine has three major components: the first is crawling e-newspapers and extracting relevant information using NLP techniques; the second is storing the extracted information in a database and creating an index for efficient search; and the third is performing search queries using a user-friendly interface.
The authors compare their search engine to several baseline methods using a dataset of news articles. According to the results, the proposed search engine outperforms the baseline methods in terms of accuracy and efficiency.
By demonstrating the effectiveness of using NLP techniques for information extraction and news search, the paper makes an important contribution to the field of information retrieval. The proposed search engine could be used in other domains with complex and technical text, such as scientific articles and legal documents. However, the paper does not provide a detailed analysis of the proposed approach's limitations or generalizability to other contexts. Overall, the paper offers useful insights into how to use NLP techniques for intelligent news search.
Link - https://ieeexplore.ieee.org/Xplore/cookiedetectresponse.jsp

○**”Sarcasm Detection in Tweets with BERT and GloVe Embeddings”** by Akshay Khatri, et al. proposed a machine learning technique with BERT and GloVe embeddings to detect sarcasm in tweets.
The authors begin by discussing the difficulties in detecting sarcasm in text, particularly in short-form communication such as tweets. They then propose a methodology for capturing the semantic relationships between words and phrases in tweets and identifying sarcastic tweets using BERT and GloVe embeddings.
A dataset of tweets labeled as sarcastic or non-sarcastic was used in the study. Several models, including logistic regression and neural network models, were trained and evaluated using BERT and GloVe embeddings. The results show that models with BERT embeddings outperformed those with GloVe embeddings, with an F1 score of 0.76.
The authors also performed an error analysis to determine which types of tweets were difficult to classify. They discovered that tweets with negation, sarcasm within quotations, and sarcasm involving proper nouns were particularly difficult to correctly classify.
Overall, the paper provides useful insights into the efficacy of using BERT and GloVe embeddings for detecting sarcasm in tweets. The study shows that using these embeddings can improve the performance of sarcasm detection models significantly. The study, however, was limited to a single dataset of tweets, and more research is needed to assess the generalizability of the proposed methodology to other contexts.
Link - https://arxiv.org/ftp/arxiv/papers/2006/2006.11512.pdf

○**"NLP-based Intelligent News Search Engine Using Information Extraction from e-Newspapers"** by Pooja Sharma and S.K. Soni's paper proposes a news search engine that uses natural language processing (NLP) techniques and information extraction from electronic newspapers to improve search results.
The paper discusses the problems with traditional keyword-based search engines and proposes a solution that uses natural language processing (NLP) techniques to extract relevant information from news articles. To extract meaningful information from news articles, the proposed system uses named entity recognition, relation extraction, and semantic analysis. The extracted information is then used by the system to create an index for the news articles, which is used to retrieve relevant articles in response to user queries.
The authors describe the proposed system's architecture in detail, including the various modules used for information extraction and indexing. The paper also discusses the proposed system's evaluation, which is based on precision, recall, and F1-score metrics. In terms of retrieval accuracy, the authors claim that their system outperforms traditional keyword-based search engines.

Overall, the paper presents a novel approach to news search that makes use of natural language processing (NLP) techniques for information extraction and indexing. The proposed system has the potential to enhance search results and provide users with more relevant news articles. However, the system's evaluation is limited to a small set of news articles, and further testing on a larger dataset would be required to validate the effectiveness of the proposed approach.
Link - https://www.iaeng.org/publication/IMECS2013/IMECS2013_pp380-384.pdf

○"**PASSAGE RE-RANKING WITH BERT"** by Rodrigo Nogueira and Kyunghyun Cho technique for re-ranking passages in information retrieval systems using BERT (Bidirectional Encoder Representations from Transformers) models is presented in the work.
The relevance of re-ranking passages to increase the precision of information retrieval systems is emphasized in the paper, which also covers the difficulties in creating efficient re-ranking models.
Proposed Method: In the suggested method, the query and candidate passages are encoded using a BERT model, and a relevance score is then calculated between the query and each passage. The texts are then re-examined in terms of their relevance to the query using the relevance scores.
Evaluation: The authors assess their method using two common datasets and contrast the outcomes with a number of benchmarks.The The results of the evaluation show that the proposed approach significantly improves accuracy relative to baseline models and surpasses state-of-the-art passage reclassification methods.
The paper concludes that BERT-based passage re-ranking can be an effective technique for improving the accuracy of information retrieval systems and has potential applications in various domains such as search engines, question answering systems, and chatbots.
This paper makes an important contribution to information research by adapting BERT as a re-ranker passage. BERT-based models surpass the previous state-of-the-art models by a large margin. This will help the user to give them relevant documents according to their query.
Link - https://arxiv.org/pdf/1901.04085.pdf

- **Updated baseline results (system/prototype):**

➢ The updated model is a Wikipedia search engine using BERT. The system was able to improve the performance of the baseline model by incorporating semantic understanding of the text.
➢ The BERT-based search engine was able to retrieve more relevant pages compared to the baseline model.

➢ The evaluation metrics we have used here are Precision, Recall, F1, Accuracy, MAP score and MRR(Mean Reciprocal Rank). We have used evaluation metrics for a particular set of queries.

➢ **Results comparison:**

○ **Comparing f1-score, precision and recall for the queries:**

- Precision is a measure of fraction of retrieved docs that are relevant = P(relevant|retrieved)
- Recall is the fraction of relevant docs that are retrieved = P(retrieved|relevant)
- f1-Score is the harmonic mean of a system's precision and recall values.

■ Query-1: Narendra Modi

■ Query-2: Indraprastha Institute of Technology

➔ **Using BERT:**

```
0.5
                                          query  precision  recall  \
0                                 Narendra Modi        0.4     1.0
1  Indraprastha Institute of Information Technology       0.2     0.5

   f1_score
0  0.571429
1  0.285714
```

➔ **Using TF-IDF:**

```
Query: Narendra Modi
Expected results: Narendra Modi, PM Narendra Modi
Top TF-IDF results: Narendra Modi, Narendra Modi Stadium, PM Narendra Modi, Premiership of Narendra Modi, Jashodaben Modi, Public image of Narendra Modi, List of international prime mi
Precision: 0.20, Recall: 1.00, F1: 0.33

Query: Indraprastha Institute of Information Technology
Expected results: Indraprastha Institute of Information Technology, Delhi, Higher Education
Top TF-IDF results: Indraprastha Institute of Information Technology, Delhi, Guru Gobind Singh Indraprastha University, List of institutions of higher education in Delhi, IIT Delhi, Ed
Precision: 0.10, Recall: 0.50, F1: 0.17

Average Precision: 0.15, Average Recall: 0.75, Average F1: 0.25
```

○ **Comparing accuracy for the queries:**

- Accuracy is expressed as a percentage of the number of relevant documents retrieved out of the total number of documents retrieved.

■ Query-1: Narendra Modi

■ Query-2: Indraprastha Institute of Technology

➔ **Using BERT:**

```
Test query 1: Narendra Modi
Expected titles: ['Narendra Modi', 'PM Narendra Modi']
Search titles: ['PM Narendra Modi', 'Jashodaben Modi', 'Narendra Modi', 'Narendra Modi Stadium', 'Premiership of Narendra Modi']
Accuracy: 1.0

Test query 2: Indraprastha Institute of Information Technology
Expected titles: ['Indraprastha Institute of Information Technology, Delhi', 'Higher Education']
Search titles: ['Indraprastha Institute of Information Technology, Delhi', 'Guru Gobind Singh Indraprastha University', 'Education in Delhi', 'List of colleges affiliated with Guru Got
Accuracy: 0.5
```

➔ **Using TF-IDF:**

```
Expected titles: ['Narendra Modi', 'PM Narendra Modi']
Search titles: ['1. A', '2. The', '3. This', '4. This', '5. PM']
Accuracy: 0.0

Expected titles: ['Indraprastha Institute of Information Technology, Delhi', 'Higher Education']
Search titles: ['1. Institute', '2. There', '3. This', '4. Tejendra', '5. This']
Accuracy: 0.0
```

○ **Comparing Average Precision and Average MAP for the queries:**

- AP (Average precision) the average precision is the mean of the precision scores after each relevant document is retrieved.
- mAP (mean average precision) is the average of AP

■ Query-1: Narendra Modi

■ Query-2: Indraprastha Institute of Technology

➔ **Using BERT:**

```
Test query 1: Narendra Modi
Expected titles: ['Narendra Modi', 'PM Narendra Modi']
Search titles: ['PM Narendra Modi', 'Jashodaben Modi', 'Narendra Modi', 'Narendra Modi Stadium', 'Premiership of Narendra Modi']
Average precision: 0.8333333333333333

Test query 2: Indraprastha Institute of Information Technology
Expected titles: ['Indraprastha Institute of Information Technology, Delhi', 'Higher Education']
Search titles: ['Indraprastha Institute of Information Technology, Delhi', 'Guru Gobind Singh Indraprastha University', 'Education in Delhi', 'List of colleges affiliated with Guru Got
Average precision: 0.5

Average MAP: 0.6666666666666666
```

➔ **Using TF-IDF:**

```
Test query 5: Narendra Modi
Expected titles: ['Narendra Modi', 'PM Narendra Modi']
Search titles: ['1. The', '2. This', '3. This', '4. PM', '5. Jashodaben']
Average precision: 0.0

Test query 5: Indraprastha Institute of Information Technology
Expected titles: ['Indraprastha Institute of Information Technology, Delhi', 'Higher Education']
Search titles: ['1. Institute', '2. There', '3. This', '4. Tejendra', '5. This']
Average precision: 0.0

Average MAP: 0.0
```

○ **Comparing MRR for the queries:**

- MRR (Mean Reciprocal Rank) is superior to other evaluation metrics because it takes into consideration the position of the relevant document in the list of documents that were retrieved, which is crucial to the user's perception of the system.
- MRR additionally tracks how quickly a user locates relevant information, whereas other evaluation metrics do not.

■ Query-1: Narendra Modi

■ Query-2: Indraprastha Institute of Technology

➔ **Using BERT:**

```
Test query 1: Narendra Modi
Expected titles: ['Narendra Modi', 'PM Narendra Modi']
Search titles: ['PM Narendra Modi', 'Jashodaben Modi', 'Narendra Modi', 'Narendra Modi Stadium', 'Premiership of Narendra Modi']
MRR: 1.0

Test query 2: Indraprastha Institute of Information Technology
Expected titles: ['Indraprastha Institute of Information Technology, Delhi', 'Higher Education']
Search titles: ['Indraprastha Institute of Information Technology, Delhi', 'Guru Gobind Singh Indraprastha University', 'Education in Delhi', 'List of colleges affiliated with Guru Gob
MRR: 1.0

Average MRR: 1.0
```

➔ **Using TF-IDF:**

```
Test query 5: Narendra Modi
Expected titles: ['Narendra Modi', 'PM Narendra Modi']
Search titles: ['1. The', '2. This', '3. PM', '4. Jashodaben', '5. The']
MRR: 0.0

Test query 5: Indraprastha Institute of Information Technology
Expected titles: ['Indraprastha Institute of Information Technology, Delhi', 'Higher Education']
Search titles: ['1. Institute', '2. There', '3. This', '4. Tejendra', '5. This']
MRR: 0.0

Average MRR: 0.0
```

- **The proposed method (features/ data analysis)**:

➢ Data Collection: The data is collected using the Wikipedia API. The search query and language are given as input to the API, and it returns the search results as JSON data.

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting wikipedia
  Downloading wikipedia-1.4.0.tar.gz (27 kB)
  Preparing metadata (setup.py) ... done
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.9/dist-packages (from wikipedia) (4.11.2)
Requirement already satisfied: requests<3.0.0,>=2.0.0 in /usr/local/lib/python3.9/dist-packages (from wikipedia) (2.27.1)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.9/dist-packages (from requests<3.0.0,>=2.0.0->wikipedia) (2022.12.7)
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/python3.9/dist-packages (from requests<3.0.0,>=2.0.0->wikipedia) (2.0.12)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.9/dist-packages (from requests<3.0.0,>=2.0.0->wikipedia) (3.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.9/dist-packages (from requests<3.0.0,>=2.0.0->wikipedia) (1.26.15)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.9/dist-packages (from beautifulsoup4->wikipedia) (2.4)
Building wheels for collected packages: wikipedia
  Building wheel for wikipedia (setup.py) ... done
  Created wheel for wikipedia: filename=wikipedia-1.4.0-py3-none-any.whl size=11696 sha256=4a139d74728c3a2d25e85e5795c039cbfaabaf5d93445936af8d6b865fa1814c
  Stored in directory: /root/.cache/pip/wheels/c2/46/f4/caa1bee71096d7b0cdca2f2a2af45cacf35c5760bee8f00948
Successfully built wikipedia
Installing collected packages: wikipedia
Successfully installed wikipedia-1.4.0
```

➢ Data Preprocessing: The text data obtained from the API is preprocessed by removing the HTML tags, punctuations, stop words, and numbers. The text data is then tokenized into words.In general, for BERT-based models, the data preprocessing step typically involves tokenization of the input text], and converting the text into numerical input embeddings that can be fed into the BERT model. This can be done using the tokenizer provided by the transformers library in Python.

➢ Embedding Generation: The preprocessed text data is fed into the BERT model to generate contextualized word embeddings. These embeddings capture the contextual meaning of the words in the documents, which is useful for measuring their similarity.

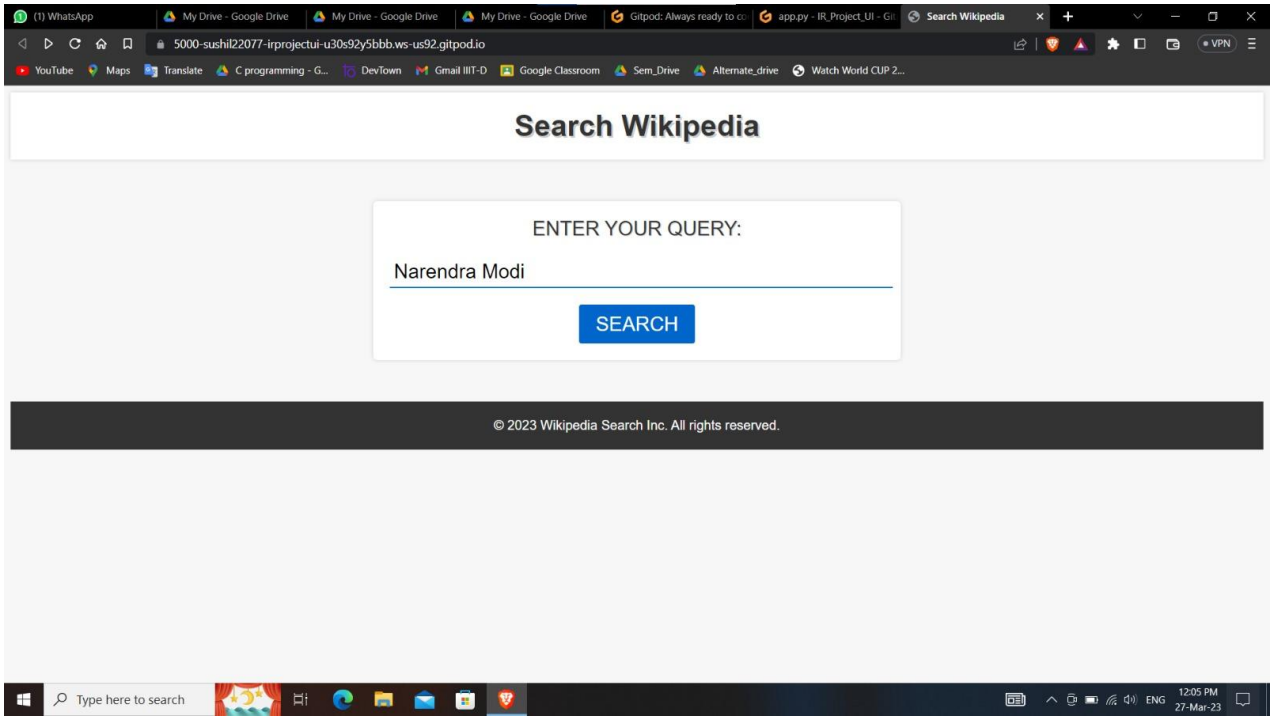
```
# Returns the BERT embeddings for the given text
def get_bert_embeddings(text):

    input_ids = torch.tensor(tokenizer.encode(text, add_special_tokens=True, max_length = 512)).unsqueeze(0)
    input_ids = input_ids.to(device)
    outputs = model(input_ids)
    last_hidden_state = outputs.last_hidden_state
    embeddings = torch.mean(last_hidden_state, dim=1).squeeze()
    return embeddings.detach().cpu().numpy()
```

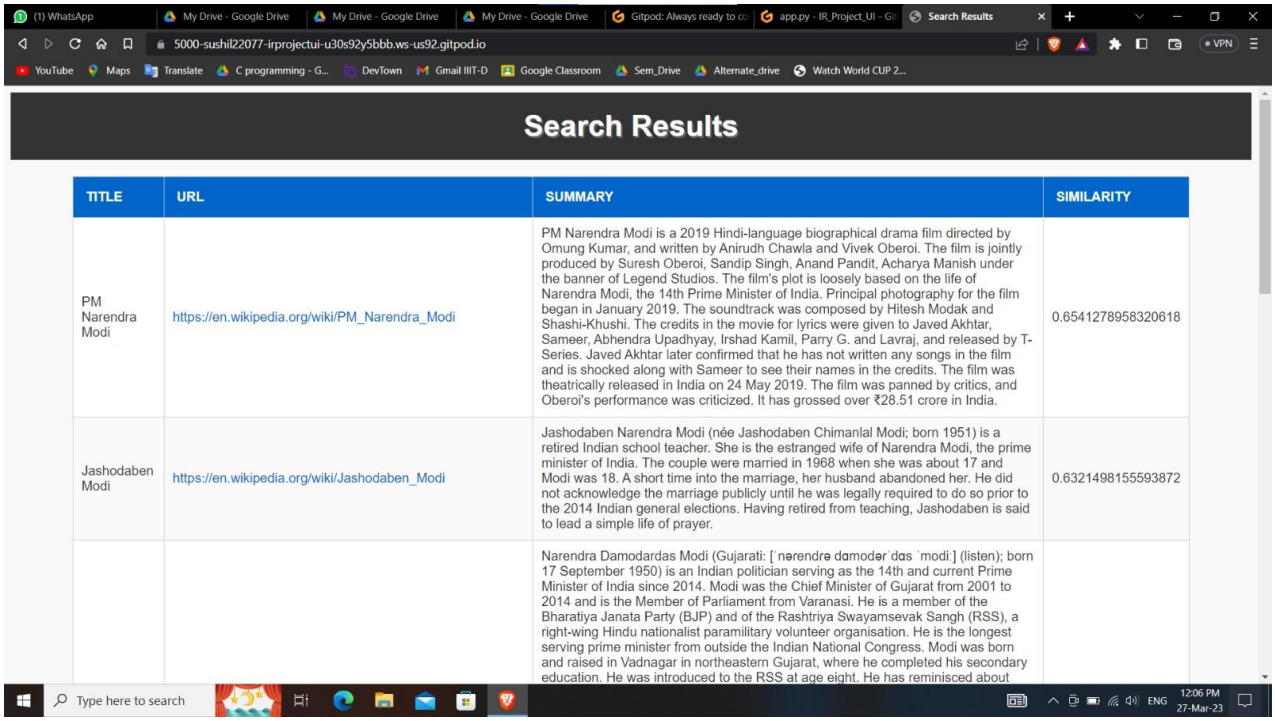
➢ Search using the Embeddings: The search query is also fed into the BERT model to generate an embedding for it. The cosine similarity is calculated between the query embedding and the embeddings of each document. The documents with the highest cosine similarity scores are considered the most relevant and are returned as search results.

```python
# Returns the top 'n_results' search results from Wikipedia for the given query
def search_wikipedia(query, n_results=5, similarity_threshold = 0.5 ):

    # get search results from Wikipedia API
    search_results = wikipedia.search(query, results=n_results)

    # initialize list to store results
    results = []

    # iterate over search results and get BERT embeddings for each page summary
    try:
      for result in search_results:
          try:
              # get page summary
              page = wikipedia.page(result)
              summary = page.summary

              # get BERT embeddings for query and page summary
              query_embeddings = get_bert_embeddings(query)
              summary_embeddings = get_bert_embeddings(summary)

              # calculate cosine similarity between query and page summary embeddings
              similarity = 1 - (distance.cosine(query_embeddings, summary_embeddings))

              # filter out results with similarity score below threshold
              if similarity < similarity_threshold:
                  continue
              # add result to list
              results.append({'title': page.title, 'url': page.url, 'summary': summary, 'similarity': similarity})
          except wikipedia.exceptions.DisambiguationError as e:
              # if page is disambiguation page, skip it
              continue
    except:
      print("Page Not Found")

    # sort results by similarity in descending order
    results = sorted(results, key=lambda x: x['similarity'], reverse=True)

    # return top n_results results
    return results[:n_results]
```

➢ Evaluation Metrics: To evaluate the performance of the search engine, we calculate the Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR) for the top k search results. MAP measures the average precision of the search engine across multiple queries, while MRR measures the rank of the first relevant result.

➢ Results Display: The search results are displayed in the console. The top k results are displayed, where k is a user-defined parameter.
For designing the UI, HTML, CSS and Flask have been used.





- **Work need to be done in future** :

➢ User Feedback: To improve the search results, we will ask the user to provide feedback on the relevance of the search results. The user can mark each search result as

relevant or irrelevant. The feedback is then used to re-rank or discard the irrelevant search results.